means for providing the clear portion to memory locations accessible by a processor; and

remainder memory for storing the remainder portion of the secure program, the remainder memory not directly accessible by the processor;

means for requesting subsets of the remainder portion for use by the processor; and

means, within the security chip, for checking that the requested subset is a predetermined subset ~~expected to be requested given a~~ dependent on the stored state for the processor.

2. (Once Amended Herein) The apparatus of claim 1, wherein the secure program stored in the program memory is stored with the **[program]** clear portion and the remainder portion stored separately.

3. (Unamended) The apparatus of claim 1, wherein the remainder portion is a set of branch instructions of the secure program.

4. (Unamended) The apparatus of claim 3, wherein the security chip further includes means for caching branch statements based on recently executed branches.

5. (Once Amended Herein) The apparatus of claim 1, wherein the means for decrypting **[branches]** portions of the secure program is configured with a decryption key.

6. (Once Amended Herein) The apparatus of claim 5, wherein the decryption key is stored in a volatile memory.

7. (Unamended) The apparatus of claim 6, wherein the volatile memory is distributed over the security chip, the

security chip further comprising overlying circuitry which overlies and obscures at least a part of the volatile memory.

8. (Once Amended Herein) The apparatus of claim 7, wherein the overlying circuitry is coupled to a power source for the volatile memory such that the removal of the **[overly]** overlying circuitry removes the power to the overlying circuitry.

9. (Once Amended Herein) The apparatus of claim 1, further comprising:

[a] clocking means, within the security chip, for determining a rate of instruction execution of the processor**[,]**; and

[wherein the security chip responds to] timing response means for rejecting processor requests **[only]** when the clocking means determines that the rate is **[within an expected]** outside a range of normal operation for the processor.

10. (Unamended) The apparatus of claim 1, further comprising a data decompressor for decompressing the secure program after decryption, wherein the secure program is compressed before encryption.

11. (Unamended) The apparatus of claim 10, wherein the decompressor is an entropy decoder.

12. (Once Amended Herein) The apparatus of claim 1, further comprising:

[a] checksum means, within the security chip, for determining a checksum of bus accesses on a processor bus**[,]**; and

[wherein the security chip responds to] checksum response means for rejecting processor requests **[only]** when the **[determined]** checksum **[matches an expected checksum]** does not match a predetermined checksum for those bus accesses.

13. (Unamended) The apparatus of claim 1, further comprising a data scrambler for reordering data elements of the secure program according to a reversible and deterministic pattern determined by a key value, wherein the secure program is reordered by the inverse of the data scrambler before being placed in the program memory.

14. (Unamended) The apparatus of claim 13, wherein the data scrambler comprises a plurality of first-in, first-out buffers.

15. (Unamended) The apparatus of claim 13, wherein the reversible and deterministic pattern is generated by reference to the output of a pseudorandom number generator.

16. (Once Amended Herein) The apparatus of claim 1, wherein the means for decrypting portions of the secure program operates based on the key value and the output of a pseudorandom number generator.

17. (Once Amended Herein) The apparatus of claim 1, further comprising means for altering the operation of the security chip and the program flow of the secure program when said means for checking detects that ~~an~~ **[unexpected]** ~~invalidly~~ the requested subset is not within the valid predetermined set of subsets ~~has been requested~~, **[where by]** whereby the altered operation causes a negative effect on the program flow or operation.

18. (Unamended) The apparatus of claim 17, wherein the means for altering is a means for halting the processor.

19. (Unamended) An apparatus for encrypting ~~securing~~ program data to prevent ~~from~~ unauthorized copying, comprising;

a branch separator for extracting branch statements from the program data;

a compressor for compressing the extracted branch statements and a remainder of the program data to form compressed data; and

an encryptor for encrypting the compressed data.

20. (Once Amended Herein) **[The apparatus of claim 19, wherein the branch separator comprises:]** An apparatus for ~~securing~~ encrypting program data ~~from~~ to prevent unauthorized copying, comprising;

a branch separator for extracting branch statements from the program data comprising:

means for automatically generating checksum data representing checksums of program data; and

means for automatically generating timing information used to assess timing of program data processing**[,]**;

a compressor for compressing the extracted branch statements, a remainder of the program data, **[whereby]** the checksum data, and the timing information, **[are compressed by the compressor and encrypted by the encryptor]** to form compressed data; and

an encryptor for encrypting the compressed data.

21. (Once Amended Herein) A method of executing a secure program to prevent copying of the secure program in a usable form from information acquired over an insecure processor bus, the usable form being a copy which replaces the functionality of the original, comprising the steps of:

accepting a request from **[the insecure]** a processor over the insecure processor bus for a block of program data, the block of program data including at least one of one or more program instructions or one or more program data elements;

decrypting, in a secure manner, the block of program data into a clear portion and a remainder portion;

providing the clear portion to the **[insecure]** processor over the insecure processor bus; and

accepting requests from the **[insecure]** processor over the insecure processor bus for elements of the remainder portion;

checking that the request is **[proper given]** consistent with the state of the **[insecure]** processor and previous requests;

processing the requests from the **[insecure]** processor for elements of the remainder portion; and

responding to the requests with request responses, wherein the request responses do not contain enough**[, wherein underlying remainder portion elements are not feasibly determined by reference to only the]** information content **[of a response to a request]** to recreate the remainder portion with substantially less computational effort than required to create ~~the secure program~~ said remainder portion.

22. (Once Amended Herein) The method of claim 21, further comprising the steps of:

separating a program into the clear portion and the remainder portion to form a secure program; and

encrypting the secure program prior to placing the secure program **[in a memory accessible by attackers intent on making unauthorized copies of the secure program]** into an insecure memory.

23. (Once Amended Herein) The method of claim 22, wherein the step of separating is a step of separating branch instructions of the **[secure]** program from other instructions of the **[secure]** program.

24. (Unamended) The method of claim 21, wherein the step of decrypting is performed with a decryption key.

25. (Once Amended Herein) The method of claim 24, further comprising the step of storing the decryption key in a volatile memory.

26. (Once Amended Herein) The method of claim 25, further comprising the steps of:

providing a power source to the volatile memory;

covering the volatile memory with a circuit such that the power source is removed from the volatile memory when the circuit is disturbed and **[the contents of the volatile memory cannot be easily measured without removing]** the circuit shields the volatile memory from probing.

27. (Once Amended Herein) The method of claim 21, further comprising the step of checking a rate of instruction execution of the processor prior to providing a request response **[to a request for information]**.

28. (Unamended) The method of claim 21, further comprising the step of decompressing the secure program after decryption, wherein the secure program is compressed before encryption.

29. (Once Amended Herein) The method of claim 21, further comprising the steps of:

determining a checksum of bus accesses on a processor bus;

comparing the checksum to a precalculated checksum **[expected]** for a set of **[the]** instructions of the secure program which **[were expected to be executed]** are executed under normal operation; and

preventing the unobstructed operation of the secure program when the checksum and the precalculated checksum differ.

30. (Once Amended Herein) The method of claim 21, further comprising **[a]** the steps of:

scrambling an order of data elements of the secure program according to a reversible and deterministic pattern determined by a key value prior to storage in **[a memory accessible by attackers]** an insecure memory; and

descrambling the order of the data elements upon proper request of the processor.

31. (Unamended) The method of claim 30, wherein the step of scrambling comprises a step of generating a pseudorandom number used to form the reversible and deterministic pattern.

32. (Once Amended Herein) A method for ~~securing~~ encrypting a program ~~against~~ to prevent unauthorized copying, comprising the steps of:

separating program code according to sequences of nonbranch instructions and branch instructions;

compressing the **[non-branch]** nonbranch instructions to form a first set of compressed data;

compressing the branch instructions to form a second set of compressed data; and

encrypting the first and second sets of compressed data.

Please enter new claim 33 as follows:

--33. (NEW) An apparatus for executing a secure program in an insecure computer system, wherein the ability to make workable copies of the secure program during execution of the secure program using the insecure computer system is inhibited, a workable copy being a copy which replaces the functionality of the original secure program, the apparatus comprising: